

APPLICATION
FOR
UNITED STATES LETTERS PATENT
ENTITLED

FLOW SCHEDULING FOR NETWORK APPLICATION APPARATUS

TO WHOM IT MAY CONCERN:

BE IT KNOWN THAT (1) JC FERGUSON and (2) YEVGENY KORSUNSKY
of (1) 36 Willow Road, Harvard, Worcester County, Massachusetts,
01451 and (2) 20 Hayden Lane, Bedford, Middlesex County,
Massachusetts, 01730, invented certain new and useful
improvements entitled as set forth above of which the following
is a specification:

PATENT GROUP
FOLEY, HOAG & ELIOT LLP
ONE POST OFFICE SQUARE
BOSTON, MA 02109-2170
TEL: 617-832-1000
FAX: 617-832-7000

1 Docket No.: CBM-001.02

2

3 FLOW SCHEDULING FOR NETWORK APPLICATION APPARATUS

4

5 CLAIM OF PRIORITY

6 This application claims priority to United States Provisional
7 Application Number 60/235,281, entitled "Optical Application
8 Switch Architecture with Load Balancing Method", and filed on
9 September 25, 2000, naming Mike Ackerman, Stephen Justus, Throop
10 Wilder, Kurt Reiss, Rich Collins, Derek Keefe, Bill Terrell, Joe
11 Kroll, Eugene Korsunky, A.J. Beaverson, Avikudy Srikanth, Luc
12 Parisean, Vitaly Dvorkian, Hung Trinh, and Sherman Dmirty as
13 inventors, the contents of which are herein incorporated by
14 reference.

15

16 BACKGROUND OF THE INVENTION

17 (1) Field of the Invention

18 The present invention relates generally to increased
19 efficiency of data flow processing, and more particularly to
20 improved flow scheduling methods and systems for multiple
21 processors.

22 (2) Description of the Prior Art

23 Increasing numbers of businesses, services, and other
24 providers are expanding their offerings on the internet. The

1 basic structure for providing network services, however, is
2 constrained with data transport dependencies. Unfortunately, a
3 given service is often provided from a single network location
4 that is deemed the central location for the service. This
5 location may be identified by a destination internet protocol (IP)
6 address that corresponds to a server that is capable of receiving
7 and processing the request. Prior art systems attempt to ease the
8 demand for a given service by providing a multiplicity of servers
9 at the destination IP address, wherein the servers are managed by
10 a content-aware flow switch. The content-aware flow switch
11 intercepts requests for the application or service and preferably
12 initiates a flow with a server that maintains a comparatively low
13 processing load. The prior art systems therefore include methods
14 for communicating a client request to a best-fit server, wherein
15 the best-fit server can be identified using server metrics that
16 include information related to the current load and recent
17 activity of the servers, network congestion between the client and
18 the servers, and client-server proximity information. In some
19 systems, the distance between client and server can be great as
20 measured geographically and/or via network hops, etc., and such
21 information can be a factor in selecting the best-fit server. In
22 some methods and systems, a obtaining server loading information
23 includes a processing known as "pinging", a technique that can
24 often be inaccurate.

1 There is currently not a system or method that provides
2 accurate and reliable information regarding processor loading and
3 other factors essential to determining a best-fit processor.

4 What is needed is a system and method that utilizes intrinsic
5 rather than extrinsic data from a multiplicity of processors to
6 determine an efficient algorithm for distributing flows to the
7 processors.

8

9 SUMMARY OF THE INVENTION

10 The methods and systems of this invention provide a scalable
11 architecture and method to facilitate the allocation of network
12 services and applications by distributing the services and
13 applications throughout a network such as the internet. In an
14 embodiment, the methods and systems can be implemented using a
15 switch architecture that can include applications processors that
16 can execute applications and services according to subscriber
17 profiles. In one embodiment, the applications processors utilize
18 the LINUX operating system to provide an open architecture for
19 downloading, modifying, and otherwise managing applications. The
20 switch architecture can also include a front-end processor that
21 interfaces to the network and the application processors,
22 recognizes data flows from subscribers, and distributes the data
23 flows from the network to the applications processors for
24 applications processing according to subscriber profiles. In an

1 embodiment, the front-end processors can recognize data flows from
2 non-subscribers, and switch such data flows-to an appropriate
3 destination in accordance with standard network switches. In one
4 embodiment, the front-end processors include flow schedules for
5 distributing subscriber flows amongst and between several
6 applications processors based on existing flow processing
7 requirements, including for example, policy.

8 In an embodiment, the applications processors and front-end
9 processors can be connected to a control processor that can
10 further access local and remote storage devices that include
11 subscriber profile information and applications data that can be
12 transferred to the front-end or applications processors. The
13 control processor can further aggregate health and maintenance
14 information from the applications and front-end processors, and
15 provide a communications path for distributing health,
16 maintenance, and/or control information between a management
17 processor and the front-end and applications processors.

18 In an embodiment, the methods and systems disclosed herein
19 can include the functionality of a switch that can be located at
20 the front-end of a network of servers, while in another
21 embodiment, the network apparatus may be between routers that
22 connect networks.

23 In one embodiment, the front-end processors can be Network
24 Processor Modules (NPMs), while the at least one applications

1 processor can be Flow Processor Modules (FPMs). The control
2 processor can include a Control Processor Module (CPM). In this
3 embodiment, the NPMs can interface to a communications system
4 network such as the internet, receive and classify flows, and
5 distribute flows to the FPMs according to a flow schedule that can
6 be based upon FPM utilization. The at least one FPM can host
7 applications and network services that process data from
8 individual flows using one or more processors resident on the
9 FPMs. The CPM can coordinate the different components of the
10 switch, including the NPMs and FPMs, allow management access to
11 the switch, and support access to local storage devices. Local
12 storage devices can store images, configuration files, and
13 databases that may be utilized when applications execute on the
14 FPMs.

15 In an embodiment, the methods and systems of the invention
16 can also allow the CPM to access a remote storage device that can
17 store applications and databases. An interface to at least one
18 management server (MS) module can receive and aggregate health and
19 status information from the switch modules (e.g., NPMs, FPMs,
20 CPMs) through the CPMs. In one embodiment, the MS module can
21 reside on a separate host machine. In another embodiment, the
22 management server module functionality can be incorporated in a
23 processor resident on a CPM.

1 In one embodiment, an internal switched Ethernet control bus
2 connects the internal components of the switch and facilitates
3 management and control operations. The internal switched Ethernet
4 control bus can be separate from a switched data path that can be
5 used for internal packet forwarding.

6 In an embodiment of the invention, the NPMS, the CPMs, the
7 FPMs, and the interconnections between the NPMS, CPMs, and FPMs,
8 can be implemented with selected redundancy to enhance the fault
9 tolerant operations and hence system reliability. For example, in
10 one embodiment wherein two NPMS, ten FPMs, and two CPMs can be
11 implemented, the two NPMS can operate in redundant or
12 complementary configurations. Additionally, the two CPMs can
13 operate in a redundant configuration with the first CPM
14 operational and the second CPM serving as a backup. The NPMS and
15 CPMs can be controlled via the Management Server module that can
16 determine whether a particular NPM or CPM may be malfunctioning,
17 etc. In this same example, up to two FPMs can be identified as
18 reserve FPMs to assist in ensuring that, in case of an FPM
19 failure, eight FPMs can function at a given time, although those
20 with ordinary skill in the art will recognize that such an example
21 is provided for illustration, and the number of reserve or
22 functioning FPMs can vary depending upon system requirements, etc.
23 The illustrated FPMs can be configured to host one or more
24 applications, and some applications can be resident on multiple

1 FPMs to allow efficient servicing for more heavily demanded
2 applications. Data flows entering the switch in-this
3 configuration can be received from an originator, processed by a
4 NPM and returned to the originator, processed by a NPM and
5 forwarded to a destination, forwarded by a NPM to a flow processor
6 and returned via the NPM to the originator, or forwarded by a NPM
7 to a flow processor and forwarded by the NPM to a destination. In
8 an embodiment wherein two or more NPMs are configured for
9 complementary operation, a flow received by a first NPM may be
10 processed, forwarded to a second NPM, and forwarded by the second
11 NPM to a destination. In another embodiment, the first NPM can
12 receive a flow and immediately forward the flow to the second NPM
13 for processing and forwarding to a destination. In complementary
14 NPM embodiments, FPM processing can also be included within the
15 described data paths.

16 In an embodiment, the well-known Linux operating system can
17 be installed on the FPM and CPM processors, thereby providing an
18 open architecture that allows installation and modification of,
19 for example, applications residing on the FPMs. In an embodiment,
20 the NPMs can execute the well-known VxWorks operating system on a
21 MIPS processor and a small executable on a network processor.

22 The methods and systems herein provide a flow scheduling
23 scheme to optimize the use of the applications processors. In an
24 embodiment, the applications processors can be understood as

1 belonging to a group, wherein the applications processors within a
2 given group are configured identically. Flow scheduling can be
3 performed and adapted accordingly for the different groups.

4 In one embodiment, applications processors from a given group
5 can report resource information to the control processors at
6 specified intervals. The resource information can include
7 intrinsic data from the applications processors such as CPU
8 utilization, memory utilization, packet loss, queue length or
9 buffer occupation, etc. The resource information can be provided
10 using diagnostic or other applications processor-specific
11 information.

12 The control module can process the resource information for
13 the applications processor(s) of a given group, and compute a flow
14 schedule vector based on the resource information, wherein in some
15 embodiments, current resource information can be combined with
16 historic resource information to compute the flow schedule vector.
17 The flow schedule vector can be provided to the front-end
18 processors and thereafter utilized by the front-end processors to
19 direct flows to the various applications processors. For example,
20 a front-end processor can identify a flow and the request
21 associated therewith, identify the group of applications
22 processors configured to process the flow/request, and thereafter
23 consult a corresponding flow scheduling vector to determine that

1 applications processor for which the flow/request should be
2 directed for processing.

3 Other objects and advantages of the invention will become
4 obvious hereinafter in the specification and drawings.

5

6 BRIEF DESCRIPTION OF THE DRAWINGS

7 A more complete understanding of the invention and many of
8 the attendant advantages thereto will be readily appreciated as
9 the same becomes better understood by reference to the following
10 detailed description when considered in conjunction with the
11 accompanying drawings, wherein like reference numerals refer to
12 like parts and wherein:

13 FIG. 1A shows four example modes of operation for the network
14 apparatus disclosed herein;

15 FIG. 1B shows an illustration of an edge-based firewall
16 embodiment for the systems and methods disclosed herein;

17 FIG. 2 is a block diagram of an apparatus according to the
18 invention;

19 FIG. 3A is a block diagram of the basic data flow through the
20 apparatus of FIG. 2;

21 FIG. 3B is a block diagram of a storage area network
22 embodiment for the apparatus of FIG. 2;

23 FIG. 4 is a diagram of a redundant architecture for a system
24 according to FIG. 2;

1 FIG. 5 is a schematic of a Network Processor Module (NPM) for
2 the systems of FIGS. 2 and 4;

3 FIGS. 6A, 6B, 6C, 6D, 6E, and 6F detail embodiments of a
4 network interface for the NPM of FIG. 5;

5 FIG. 7 illustrates a crossover on the backplane within the
6 illustrated NPM of FIG. 5;

7 FIG. 8 is an architectural block diagram of a Flow Processor
8 Module (FPM) for the embodiments of FIGS. 2 and 4;

9 FIG. 9 is a block diagram of an illustrative Control
10 Processor Module (CPM) architecture according to the
11 representative systems of FIGS. 2 and 4; and,

12 FIG. 10 is a block diagram illustrating a logic flow for flow
13 scheduling for the methods and systems of FIGS. 2-4.

14
15 DESCRIPTION OF ILLUSTRATED EMBODIMENTS

16 To provide an overall understanding of the invention, certain
17 illustrative embodiments will now be described; however, it will
18 be understood by one of ordinary skill in the art that the systems
19 and methods described herein can be adapted and modified to
20 provide systems and methods for other suitable applications and
21 that other additions and modifications can be made to the
22 invention without departing from the scope hereof.

23 For the purposes of the disclosure herein, an application can
24 be understood to be a data processing element that can be

1 implemented in hardware, software, or a combination thereof,
2 wherein the data processing element can include a number of states
3 that can be zero or any positive integer.

4 For the purposes of the methods and systems described herein,
5 a processor can be understood to be any element or component that
6 is capable of executing instructions, including but not limited to
7 a Central Processing Unit (CPU).

8 The invention disclosed herein includes systems and methods
9 related to a network apparatus that can be connected in and
10 throughout a network, such as the internet, to make available
11 applications and services throughout the network, to data flows
12 from subscriber users. Although the apparatus can perform the
13 functions normally attributed to a switch as understood by one of
14 ordinary skill in the art, and similarly, the apparatus can be
15 connected in and throughout the network as a switch as understood
16 by one of ordinary skill in the art, the apparatus additionally
17 allows the distribution of applications throughout the network by
18 providing technical intelligence to recognize data flows received
19 at the switch, recall a profile based on the data flow, apply a
20 policy to the data flow, and cause the data flow to be processed
21 by applications or services according to the profile and/or
22 policy, before forwarding the data flow to a next destination in
23 accordance with switch operations as presently understood by one
24 of ordinary skill in the art. In an embodiment, the next

1 destination may be a network address or a another device otherwise
2 connected to the network apparatus. By increasing the
3 availability of services by distributing the services throughout
4 the network, scalability issues related to alternate solutions to
5 satisfy increased demand for applications and services, are
6 addressed.

7 FIG. 1A displays four exemplary modes and corresponding
8 illustrative examples of operation for the network apparatus or
9 device presented herein, wherein such modes are provided for
10 illustration and not limitation. The first mode shown in FIG. 1A
11 can be utilized for, as an example, a firewall application,
12 wherein data flows can be received by the network apparatus and
13 processed in what can otherwise be known as a "pass or drop"
14 scenario. In such applications, the network apparatus can accept
15 data flows from one interface and either pass the flow to a
16 destination using a second interface according to permissions
17 provided by the firewall, or the data flow may be dropped (i.e.,
18 not forwarded to the destination). In the second scenario of FIG.
19 1A, labeled "modify, source, and send," a data flow received by
20 the network apparatus can be received by a first interface,
21 modified, and forwarded via a second interface to a destination.
22 An example embodiment of the second scenario includes content
23 insertion. In the third scenario of FIG. 1A, the network
24 apparatus can function as a proxy wherein data flows can be

1 received, processed, and returned at a first data interface, and
2 similarly, data flows received from a second data interface can be
3 processed and returned via the second interface, wherein the
4 respective data flows can be dependent or otherwise related.
5 Sample embodiments of the third scenario of FIG. 1A include
6 transaction services and protocol translation. In the fourth
7 sample embodiment of FIG. 1A, the network apparatus can be
8 utilized for applications including, for example, VoIP
9 conferencing, content insertion, and application caching, wherein
10 data flows can be received at a first interface, processed, and
11 returned via the first interface.

12 FIG. 1B provides another illustration of the network
13 apparatus and demonstrates a data flow for an edge-based firewall
14 embodiment 200 incorporating the network apparatus according to
15 the methods and systems disclosed herein. In the illustration,
16 data flows in the form of internet requests from a subscriber to
17 Internet Service Provider (ISP) A 202 and a subscriber to ISP B
18 204 are input to a Digital Subscriber Line Access Multiplexer
19 (DSLAM) 206 and thereafter forwarded to an Asynchronous Transfer
20 Mode (ATM) switch 208 within an ISP A-related Super-POP, that
21 aggregates the flows and forwards the flows to a router 210. The
22 router 210 directs the data flow traffic to the network device or
23 apparatus 12 that recognizes the flows from the respective ISP
24 subscribers 202, 204 and applies respective firewall policies. In

1 the illustrated embodiment, ISPs A and B are subscribers to the
2 network apparatus 12 and in accordance therewith, provide profiles
3 and applications/services in accordance with such profiles for
4 distribution and processing by the apparatus in conformance with
5 the profiles. In the illustrated embodiment, applications in
6 addition to the respective firewall policies, for example, can be
7 applied to the respective data flows. After the respective
8 processing is performed by the network apparatus 12, in the
9 illustrated embodiment, the data flow from the ISP A subscriber
10 202 is forwarded to the internet 212 with the applications applied
11 to the data, while the data flow from the ISP B subscriber 204 is
12 forwarded to ISP B 214 with the policy applied to the data.

13 The network apparatus 12 can also recognize data as not
14 otherwise belonging to a subscriber and therefore not eligible for
15 applications processing, wherein such data can be switched to a
16 destination in accordance with a switch presently known to one of
17 ordinary skill in the art. Those with ordinary skill in the art
18 will also recognize that although this disclosure presents the
19 apparatus connected within the network known as the internet, the
20 internet application is presented for illustration and not
21 limitation. In an embodiment wherein the apparatus is used with a
22 communications system such as the internet, the apparatus can be
23 connected at the front-end of a server network, or alternately,

1 between routers that connect networks, although the apparatus
2 disclosed herein is not limited to such embodiments.

3 FIG. 2 shows another illustrative block diagram 10 of the
4 network apparatus 12 that can host applications and connect into
5 and throughout the infrastructure of a network such as the
6 internet, thereby distributing the hosted applications and/or
7 services accordingly throughout the network. Those with ordinary
8 skill in the art will recognize that the FIG. 2 illustration is
9 intended to facilitate the disclosure of the invention and is not
10 intended as a limitation of the invention. As indicated by FIG.
11 2, the illustrated apparatus 12 includes two Network Processor
12 Module (NPMs) 14 that facilitate the flow of network into and out
13 of the network apparatus 12 by independently maintaining, in the
14 illustrated embodiment, two Gigabit Ethernet connections. Those
15 with ordinary skill with recognize that Gigabit Ethernet
16 connections are merely one high-speed data link, and other such
17 data links can be substituted without departing from the scope of
18 the invention. In an embodiment where the apparatus 12 is
19 inserted in-line on a trunk connecting subscribers to the internet
20 core, for example, the Gigabit Ethernet connections can optionally
21 interface to a subscriber network 16 and the internet core 18.
22 Those with ordinary skill in the art will recognize that in
23 another embodiment, a single NPM can be utilized, and the two
24 Gigabit Ethernet connections can connect to two different

1 networks, for example. Additionally, those with skill in the art
2 will recognize that for the illustrated system, the apparatus 12
3 can utilize a single bi-directional interface to connect to the
4 subscriber network 16 and internet core 18. The FIG. 2 NPMs 14
5 connect via an Ethernet through a cross-connect 20 to at least one
6 Flow Processor Modules (FPMs) 22 that apply applications and
7 services to data flows, and to at least one Control Processor
8 Module (CPM) 24 that can process data flow requests and collect
9 health and maintenance information from the NPMs 14 and FPMs 22.

10 Each illustrated NPM 14, FPM 22, and CPM 24 also connect to a
11 high-speed switching fabric that interconnects all modules and
12 allows internal packet forwarding of data flows between the NPM
13 14, FPM 22, and CPM 24 modules. The CPM 24 similarly
14 independently connects to the FPMs 22 and NPMs 14 in the
15 representative embodiment by a 100Base-T Ethernet Control Bus 26
16 that can be dual redundant internal switched 100Mbyte/second
17 Ethernet control planes. The illustrated CPMs 24 also connect to
18 a Management Server (MS) module 28 by a 100Base-T Ethernet, to a
19 local memory device 30, and to a Data Center 32 through a Gigabit
20 Ethernet connection. The MS module 28 allows for data collection,
21 application loading, and application deleting from the FPMs 22,
22 while the local memory device 30 and Data Center 32 can store data
23 related to applications or profile information. In the
24 illustrated system of FIG. 2, there are two NPMs 14, at least two

1 CPMs 24, and ten FPMs 22, although such a system is merely
2 illustrative, and those with ordinary skill in the art will
3 recognize that fewer or greater numbers of these components may be
4 utilized without departing from the scope of the invention. In
5 the illustrated system of FIG. 2, the two NPMs can operate in
6 complementary or redundant configurations, while the two CPMs can
7 be configured for redundancy.

8 As indicated, using an architecture according to the
9 principles illustrated, the apparatus 12 may be placed within the
10 normal scheme of a network such as the internet, wherein the
11 apparatus 12 may be located, for example, at the front-end of a
12 server network, or alternately and additionally, between routers
13 that connect networks. Using firmware and/or software configured
14 for the apparatus modules, the apparatus 12 can be configured to
15 provide applications to subscribers, wherein the applications can
16 include virus detection, intrusion detection, firewalls, content
17 filtering, privacy protection, and policy-based browsing, although
18 these applications are merely an illustration and are not intended
19 as a limitation of the invention herein. In one embodiment, the
20 NPMs 14 can receive data packets or flows and process such packets
21 entirely before forwarding the packets to the appropriate
22 destination. In the same embodiment, the NPMs 14 can receive and
23 forward the packets to an appropriate destination. Also in the
24 same embodiment, the NPMs 14 can recognize data packets that

1 require processing that can be performed by applications residing
2 on the FPMs 22; and in these instances, the NPMs 14 can perform
3 flow scheduling to determine which FPM 22 can appropriately and
4 most efficiently process the data, wherein the data packets or
5 flow can then be forwarded to the selected FPM 22 for processing.
6 In an embodiment, not all FPMs 22 can process all types of
7 processing requests or data packets. Additionally, to process a
8 data request, in some instances, a FPM 22 can require information
9 from the local memory device 30 or the remote memory device 32,
10 wherein the NPM 14 can direct the retrieval of storage data
11 through the CPM 24 and thereafter forward the storage data to the
12 FPM 22. An FPM 22 can thereafter transfer processed data to the
13 NPM 14 for forwarding to an appropriate destination. With the
14 apparatus 12 architecture such as that provided by FIGs. 1 and 3,
15 application service providers can more efficiently provide
16 services to subscribers by integrating and making available
17 services throughout a network such as the internet, rather than at
18 a single location that is often designated as a single IP address.

19 FIG. 3A shows a schematic of data flow through the apparatus
20 12 of FIG. 1. As FIG. 3A indicates, NPMs 14 may provide an
21 interface between the subscriber interface and the network core.
22 The FIG. 3A NPM 14 can receive data from a first interface 14a,
23 and depending on the data request, can process the data and
24 transmit the processed data using either the first interface 14a

1 or the second interface 14b. Alternately, the NPM 14 can forward
2 the received data to a FPM 22 that can thereafter return the
3 processed data to the NPM 14 for transmission or forwarding using
4 either the first interface 14a or the second interface 14b.
5 Similarly, the NPM 14 can receive data from the second interface
6 14b, process the data, and transmit the processed data using
7 either the first interface 14a or the second interface 14b.
8 Additionally, data received by the NPM 14 through the second
9 interface 14b can be forwarded to the FPMs 22 for processing,
10 wherein the FPMs 22 can return the processed data to the NPM 14
11 for transmission through either the first interface 14a or the
12 second interface 14b. In another example, data received by the
13 NPM 14 can be processed by multiple FPMs 22, wherein the data can
14 be forwarded to the multiple FPMs 22 through the NPM 14, and
15 returned to the NPM 14 for forwarding to a destination.

16 In an embodiment wherein two NPMs are configured for
17 complementary operation, data received at a first NPM can be
18 processed by the first NPM, transmitted to a second NPM, and
19 forwarded by the second NPM to a destination. Alternately, data
20 received at the first NPM can be forwarded to the second NPM,
21 processed, and forwarded to a destination accordingly. In yet
22 other scenarios, data received at either of the two NPMs can be
23 forwarded to any of the FPMs 22, processed, and returned to either
24 of the NPMs for forwarding to a destination. Those with ordinary

1 skill in the art will recognize that the examples of data movement
2 and processing entering, within, and exiting the apparatus 10 are
3 merely for illustration and not limitation, and references to the
4 first NPM and second NPM in the complementary embodiment can be
5 exchanged, for example, without departing from the scope of the
6 invention.

7 FIG. 3B shows the system of FIGs. 2 and 3A configured to
8 operate in accordance with a Storage Area Network (SAN) as is
9 commonly known in the art. In the configuration of FIG. 3B, the
10 NPM 14 and FPM 22 integration as indicated in FIG. 3A is
11 preserved, however, the NPM 14 and FPM 22 also maintain interfaces
12 to one or more storage devices 23 that can be any storage device
13 commonly known in the art, including but not limited to RAM, ROM,
14 diskettes, disk drives, ZIP drives, RAID systems, holographic
15 storage, etc., and such examples are provided for illustration and
16 not limitation. As FIG. 3B indicates, data can be received at the
17 NPM 14 and transferred directly to the storage devices 23; or,
18 data received by the NPM 14 can be forwarded to one or more FPMs
19 22 before being forwarded by the FPMs 22 to the storage devices
20 23, wherein the FPMs 22 can perform processing on the data before
21 forwarding the data to storage 23. Similarly, in the FIG. 3B
22 configuration, data can be retrieved from storage 23 by either the
23 NPM 14 or FPMs 22. In the FIG. 3B configuration, the NPM 14 and

1 FPMs 22 maintain external interfaces that can accommodate data
2 input and output.

3 FIG. 4 illustrates an alternate representation of the FIG. 2
4 system that implements a dual redundant architecture. In the FIG.
5 4 embodiment of a redundant architecture, there are two NPMs 14a,
6 14b, two CPMs 24a, 24b, and ten FPMs 22a-22n that reside in a
7 fourteen rack chassis. In the FIG. 4 system, eight FPMs 22 are
8 provided for typical apparatus 12 operation, with two FPMs 22
9 provided as alternates in the case of failure of up to two of the
10 operational eight FPMs 22. As FIG. 4 indicates, redundant
11 internal switched 100Mbyte/second (100Base-T) Ethernet control
12 planes 170a, 170b, provide connections between each of the NPMs
13 14a, 14b, CPMs 24a, 24b, and FPMs 22a-22n. The illustrated system
14 also includes dual fabric links 172a, 172b, wherein each FPM 22a-
15 22n and CPM 24a, 24b connect to each fabric link 172a, 172b, while
16 the first NPM 14a connects to the first fabric link 172b, and the
17 second NPM 14b connects to the second fabric link 172b to allow
18 each NPM 14a, 14b to operate independently of the other.

19 Additionally, as indicated in FIG. 4, the FIG. 4 NPMs 14a,
20 14b maintain two Gigabit Ethernet connections to the network,
21 wherein one of the connections can be to a subscriber including a
22 subscriber network, etc., while the other connection can be to the
23 internet core. Alternately, the illustrated CPMs 24a, 24b

1 maintain a Gigabit Ethernet connection to communicate with a
2 remote storage device illustrated as the data center 32 of FIG. 2.

3 FIG. 5 shows a schematic block diagram of an illustrative NPM
4 14 according to FIGs. 2 and 4. As indicated-in FIGs. 2 and 4,
5 according to the invention, the apparatus or switch 12 can include
6 one or more NPMs 14, and when more than one NPM 14 is utilized,
7 the NPMs 14 may be configured for redundant or complementary
8 operation.

9 A NPM 14 can include a modular and optional subsystem
10 illustrated in FIG. 5 as a network interface subsystem 40. This
11 subsystem 40 physically connects the switch 12 and a network,
12 thereby providing a data flow between the switch 12 and the
13 network. The NPM 14 also includes a Network Processor 42 that
14 connects to the network interface subsystem 40. The Network
15 Processor 42 can be, for example, an IQ2000 Network Processor, and
16 those with ordinary skill in the art will recognize this example
17 as an illustration and not a limitation, wherein any like device
18 performing the functions as described herein may be similarly
19 substituted. Additionally, a second processor can be co-located
20 within the NPM architecture without departing from the scope of
21 the invention. In the case of the illustrated IQ2000 Network
22 Processor 42, the network interface system 40 can connect to ports
23 A and B of the Network Processor 42 using a FOCUS bus, wherein
24 such ports shall hereinafter be referred to as FOCUS ports A and

1 B, and wherein two remaining FOCUS ports labeled C and D are
2 available on the Network Processor 42.

3 The network interface subsystem 40 can be a changeable
4 component of the NPM architecture, wherein the different options
5 can be different Printed Circuit Board (PCB) designs or pluggable
6 option boards, however, those with ordinary skill in the art will
7 recognize that such methods of implementing the network interface
8 subsystem 40 are merely illustrative and the invention herein is
9 not limited to such techniques.

10 For example, FIGs. 6A through 6F provide various illustrative
11 network interface subsystem 40 options for the FIG. 5 NPM 14.
12 Referring to FIG. 6A, the two Gigabit Ethernet interfaces 50, 52
13 to the FIG. 5 Network Processor 42 are supported through the
14 Network Processor's 42 two embedded Gigabit Ethernet Media Access
15 Control devices (MACs). In the FIG. 6A embodiment of a network
16 interface subsystem 40, the only external devices necessary for
17 Gigabit Ethernet operation include the Gigabit Ethernet physical
18 layer device (PHY) 54a, 54b and optical interfaces 56a, 56b. In
19 the illustrated embodiment, a first optical interface 56a can
20 couple to a subscriber's network equipment, while a second optical
21 interface 56b can couple to the internet core.

22 Referring now to FIG. 6B, there is an illustrative
23 configuration for the FIG. 5 NPM 14 wherein FOCUS ports A and B
24 can support up to eight 10/100 Ethernet ports through an external

1 octal 10/100 MAC 60a, 60b. In FIG. 6B, the two external eight
2 port 10/100 MACs 60a, 60b couple to the FOCUS ports and to two
3 external eight port 10/100 PHY devices 62a, 62b. The PHY devices
4 respectively couple to eight RJ-45 connections 64a, 64b. In the
5 FIG. 6B configuration, one set of eight RJ-45 ports 64a can be
6 dedicated to the subscriber's network, while the remaining eight
7 RJ-45 ports 64b can couple to the internet core. In one
8 embodiment, the architecture of FIG. 6B can allow software or
9 firmware to configure the ports as independent data streams such
10 that data received on a subscriber's port can be returned on a
11 internet port.

12 Referring now to FIG. 6C, there is a network interface
13 subsystem 40 configuration for the illustrated NPM 14 of FIG. 5,
14 wherein the switch 12 can receive ATM cells with the cooperation
15 of a Segmentation and Reassembly device (SAR) 70a, 70b connected
16 to the A and B FOCUS ports. In the configuration of FIG. 6C
17 wherein OC-3c ATM operation is illustrated, four optical
18 interfaces 72a provide the subscriber interface, while four
19 optical interfaces 72b provide the internet core interface. The
20 respective subscriber and internet optical interfaces 72a, 72b
21 couple to a four port framer 76a, 76b that provides input to a
22 Transmission SAR 70a (TX, "to" the switch 12), or receives output
23 from a Receiver SAR 70b (RX, "from" the switch 12). In the
24 illustrated configuration, the SARs 70a, 70b utilize a 32-bit SRAM

1 77 and a 64-bit SDRAM 78, although such an embodiment is merely
2 for illustration. In the illustrated system of FIG. 6C, the SAR
3 UTOPIA ports interface to the FOCUS A and B ports through a Field
4 Programmable Gate Array (FPGA) 79. Those with ordinary skill in
5 the art will recognize that the network interface subsystem of
6 FIG. 6C, as with the other diagrams provided herein, is merely
7 provided for illustration and not intended to limit the scope of
8 the invention; therefore, components may be otherwise substituted
9 to perform the same functionality, wherein for example, a single
10 SAR capable of transmission and receiving may be substituted for
11 the two SARs 70a, 70b depicted in the illustration of FIG. 6C.

12 Referring now to FIG. 6D, there is a network interface
13 subsystem 40 configuration for the illustrated NPM 14 of FIG. 4,
14 wherein OC-12c ATM operation may be enabled. In the illustrated
15 system, one OC-12c optical interface 80a can couple to the
16 subscribers, while a second OC-12c optical interface 80b can
17 couple to the internet core. In contrast to FIG. 6C, FIG. 5D
18 illustrates only a two port framer 82 that thereafter interfaces
19 to the TX and RX SARs 84a, 84b, FPGA 86, and the respective FOCUS
20 ports of the Network Processor 42.

21 Referring now to FIG. 6E, there is an OC-3C Packet Over SONET
22 (POS) configuration for the network interface subsystem 40 of FIG.
23 5. In the illustrated configuration of FIG. 6E, four optical
24 interfaces 90a can interface to the subscriber, while four optical

1 interfaces 90b can be dedicated to the internet core. The optical
2 interfaces 90a, 90b respectively couple to a four port framer 92a,
3 92b that interfaces to the A and B FOCUS ports through a FPGA 94.
4 Those with ordinary skill in the art will recognize that because
5 PPP (Point-to-Point Protocol) encapsulated packets are inserted
6 into the SONET Payload Envelope (SPE), all POS links are
7 concatenated, and the FPGA 94 utilized in FIG. 6E may therefore be
8 similar to the FPGA 86 of FIG. 6D.

9 Referring to FIG. 6F, there is a configuration of the network
10 interface subsystem 40 of FIG. 5 for a two port OC-12c POS
11 application. In the illustrated system, one optical interface
12 100a can couple to the subscriber, and another 100b can couple to
13 the internet core. The FIG. 6F optical interfaces 100a, 100b
14 couple to a two port framer 102 that interfaces to a FPGA 104 for
15 connection to the A and B FOCUS ports.

16 Referring back to FIG. 5, the illustrated Network Processor
17 42 also connects to a CPU subsystem 110 that includes a MIPS
18 processor 112 such as a QED RM700A 400 MHz MIPS processor, a
19 system controller/PCI bridge 114 such as the Galileo GT64120A
20 system controller/PC bridge, local SDRAM 116, and a Programmable
21 Logic Device (PLD) 118. In the illustrated system, the PLD 118
22 makes accessible the board specific control registers and
23 miscellaneous devices. As illustrated, the PLD 118 is connected
24 to a local high-speed bus on the GT64120A 114 with a local SDRAM

1 116, and acts as a buffer between the local high-speed bus 120 and
2 a lower speed peripheral bus 122 that has boot PROM Flash 124 and
3 non-volatile RAM (NVRAM) 126 for semi-permanent storage of
4 settings and parameters, and for providing a real-time clock for
5 time of day and date. The FIG. 5 PCI bus 127 connected to the PCI
6 bridge also includes two Fast Ethernet MACs 128a, 128b, such as
7 the Intel GD82559ER 100 Mbit MAC that includes an integrated PHY,
8 to provide redundant connections between the NPM 14 and CPM 24 via
9 a primary and secondary 100 Base-T Ethernet channel. The
10 illustrated MACs 128a, 128b reside on the PCI bus and perform
11 Direct Memory Access (DMA) transfers between the PCI internal
12 buffers and the defined buffer descriptors within the local MIPS
13 memory 112. The MACs 128a, 128b can support an unlimited burst
14 size and can be limited by PCI bridge performance. In an
15 embodiment, flow control can be utilized in a control plane
16 application to avoid unnecessary packet loss. The illustrated
17 GT64120A 114 allows the CPU 112 and other local bus masters to
18 access the PCI memory and/or device buses.

19 The FIG. 5 NPM 14 also includes a switch fabric subsystem 130
20 that provides high-speed, non-blocking data connections between
21 the NPM 14 and the other modules within the switch 12. The
22 connections include two links to another, redundant or
23 complementary NPM 14 and a link to each CPM 24. The illustrated
24 NPM's 14 portion of the fabric includes two Focus Connect devices

1 132a, 132b, wherein one Focus Connect device 132a is connected to
2 the IQ2000 42 port C using a FOCUS Bus, while another Focus
3 Connect device 132b is connected to port D.

4 In the illustrated system, the ports on the sixteen bit FOCUS
5 bus on the Focus Connect devices 132a, 132b, with the exception of
6 local port eight, are attached to a Cypress Quad Hotlink Gigabit
7 transceiver 134 that is a serial to deserial (SerDes) device 136
8 having dual redundant I/O capabilities and configured for dual
9 channel bonded mode. The dual channel bonded mode couples two
10 channels together in a sixteen-bit channel, wherein there can be
11 two such sixteen-bit channels per device. Referring now FIG. 7,
12 the dual redundant serial I/O capabilities, in cooperation with a
13 crossover on the backplane, allow any slot to be connected to any
14 other slot such that a packet or a data route vector modification
15 is not necessary when only one NPM 14 is present. The FIG. 5
16 Serdes devices 136 convert incoming serial stream data from the
17 backplane, to parallel data for forwarding to the Focus Connect
18 devices 132a, 132b. Similarly, the Serdes 136 converts parallel
19 data from the Focus Connect device 132a, 132b to serial data
20 before placing the data on the backplane.

21 For example, with the illustrated system of FIG. 4 a Focus
22 Connect device 132a, 132b is connected to the IQ2000 FOCUS C and D
23 ports and wherein the Focus Connect devices 132a, 132b maintain
24 eight ports each, in the illustrative system wherein there is a

1 fourteen slot chassis and there are ten slots for FPMs 22a-22n,
2 two slots for NPMs 14a, 14b, and two slots for CPMs 24a, 24b, the
3 Focus Connect device ports can be configured as shown in Tables 1
4 and 2:

5 Table 1
6 Focus Connect device connected to IQ2000 FOCUS Port C (132a)

Focus Connect Port	Connected Module
1	FPM, slot 1
2	FPM, slot 2
3	FPM, slot 3
4	FPM, slot 4
5	FPM, slot 5
6	CPM, slot 1
7	Other NPM, Focus Connect Port D
8	Local IQ2000, Port C

Table 2
Focus Connect device connected to IQ2000 FOCUS Port D (132b)

Focus Connect Port	Connected Module
1	FPM, slot 6
2	FPM, slot 7
3	FPM, slot 8
4	FPM, slot 9
5	FPM, slot 10
6	CPM, slot 2
7	Other NPM, Focus Connect on Port C
8	Local IQ2000, Port D

As Tables 1 and 2 indicate, using the FIG. 4 NPM 14 in a redundant system as illustrated in FIGs. 1 and 3, the dual NPMs 14a, 14b can access all FPMs 22a-22n and each CPM 24a, 24b, and vice-versa.

The fourth major subsystem of the FIG. 5 NPM 14 is a memory subsystem 140. The FIG. 5 memory subsystem is a single RAMbus channel for packet buffer storage and flow lookup table space. In the illustrated embodiment, the memory subsystem 140 includes a search processor 142 and several content addressable memories 144, although those with ordinary skill in the art will recognize that the invention herein is not limited to the memory subsystem 140 or the components thereof.

1 Referring back to FIG. 5, data received by the NPM 14 can be
2 forwarded to the IQ2000 42 that can include instructions for
3 recognizing packets or data flows. For example, CPU or processor
4 instructions can implement or otherwise utilize a hash table to
5 identify services or processing for an identified packet or flow,
6 wherein the packet or flow can subsequently be forwarded to a FPM
7 22, for example, in accordance with the service or processing.
8 Alternately, unidentified packets can be forwarded to the MIPS 112
9 that can include instructions for identifying the packet or flow
10 and associated processing or services. In an embodiment, packets
11 unable to be identified by the MIPS 112 can be forwarded by the
12 MIPS 112 to the CPM 24 that can also include instructions for
13 identifying packets or flows. Identification information from
14 either the CPM 24 or MIPS 112 can be returned to the IQ2000 42 and
15 the hash table can be updated accordingly with the identification
16 information.

17 Referring now to FIG. 8, there is a basic schematic block
18 diagram of a FPM 22 for the system illustrated in FIGs. 1-3. In
19 the embodiment of FIG. 8, the FPM 22 is based upon Intel's 440BX
20 AGPset, with a majority of the FPM functionality similar to a
21 personal computer (PC). The illustrated FPM 22 can therefore be,
22 viewed as having four main sections that include a processor or
23 CPU 120, a 440BX AGPset 122, a FOCUS interface, and peripherals.
24 In the illustrated system of FIGs. 2 and 4, the FPMs 22 are

1 identically designed, although those with ordinary skill in the
2 art will recognize that the methods and systems disclosed herein
3 may include differing FPM designs.

4 Referring to FIG. 8, the illustrated FPM 22 embodiment
5 supports a single socket 370 Intel Pentium III CPU 150 with a 100
6 Megahertz processor system bus (PSB), although such processor is
7 merely for illustration and not limitation, and those with
8 ordinary skill in the art will recognize that the invention
9 disclosed herein is not limited by the CPU selection or processor
10 component. Similarly, those with ordinary skill in the art will
11 recognize that multiple processors 150 can be incorporated within
12 the FPM architecture without departing from the scope of the
13 invention. The representative FPM 22 also includes a 440BX
14 Accelerated Graphics Port (AGPset) 152 that provides
15 host/processor support for the CPU 150.

16 Data packets moving into and out of the FPM 22 in the
17 illustrated system use a 16-bit wide 100 Megahertz bus called the
18 FOCUS bus, and in the illustrated embodiment, a full-duplex FOCUS
19 bus attaches to every FPM 22 from each NPM 14, wherein in the
20 illustrated embodiment of dual redundant NPMs 14a, 14b, every FPM
21 22 communicates with two NPMs 14a, 14b. As indicated previously,
22 the FOCUS bus signal is serialized on the NPM 14a, 14b before it
23 is placed on the backplane, to improve signal integrity and reduce
24 the number of traces. As illustrated, deserializers 154a, 154b on

1 the FPM 22 convert the signal from the backplane to a bus and the
2 bus connects the deserializers 154a, 154b to a Focus Connect 156
3 that interfaces through a FPGA 158 and Input Output Processor 160
4 to the 440BX AGPset 152. The illustrated PRC is an eight-way
5 FOCUS switch that allows the FPM 22 to properly direct packets to
6 the correct NPM 14.

7 The FIG. 8 FPM 22 also maintains peripherals including
8 control plane interfaces, mass storage devices, and serial
9 interfaces. In the illustrated FPM 22, the control plane provides
10 a dedicated path for communicating with the FPM 22 through two
11 fast Ethernet controllers 130a, 130b that interface the AGP 152 to
12 the redundant control plane. As indicated in FIGs. 2 and 4, it is
13 typically the CPM 24a, 24b that communicates with the FPM 22 via
14 the control plane. In the illustrated embodiment, the fast
15 Ethernet controllers 130a, 130b connect to control planes that are
16 switched 100 Megabits/second Ethernet networks that terminate at
17 the two CPMs 24.

18 The illustrated FPM 22 may also support different types of
19 mass storage devices that can include, for example, a M-Systems
20 DiskOnChip (DOC), a 2.5 inch disk drive, NVRAM for semi-permanent
21 storage of settings and parameters, etc.

22 Referring now to FIG. 9, there is an illustration of a sample
23 CPM 24 as presented in the systems of FIG. 2 and 4. As indicated
24 previously, the CPM 24 performs generic, switch-wide functions and

1 is connected to the other switch components through a data
2 interface that, in the illustrated embodiment, is identical to the
3 data interface of FIG. 7 for the FPM 22. Those with ordinary
4 skill in the art will recognize that the common data interfaces
5 for the FPM 22 and CPM 24 modules are merely for convenience and
6 do not limit the scope of the invention.

7 As discussed earlier, in the illustrated embodiment, the
8 control planes terminate at a CPM 24, wherein the illustrative
9 control planes are dual redundant, private, switched 100 Megabit
10 Ethernet. The switching elements are housed on the CPM 24, and
11 therefore all point-to-point connections between other modules and
12 a CPM 24 are maintained through the backplane connector.

13 Additionally, the CPM 24 controls the switch 12 boot process
14 and manages the removal and insertion of modules into the switch
15 12 while the switch 12 is operational.

16 In the illustrated CPM 24 of FIG. 9, the main CPU 170 is a
17 Pentium III processor, although the invention herein is not so
18 limited, and any processor or CPU or device capable of performing
19 the functions described herein may be substituted without
20 departing from the scope of the invention, wherein multiple
21 processors or CPUs may additionally be utilized. In the
22 illustrated CPM 24, a 440BX Accelerated Graphics Port (AGPset) 172
23 provides host/processor support for the CPU 170. The FIG. 9 AGP

1 172 supports a PCI interface to connect to miscellaneous hardware
2 devices.

3 Three fast Ethernet controllers 174a, 174b, 174c also reside
4 on the PCI bus of the 440 BX 172. One of these three fast
5 Ethernet controllers 174a provides external communications and
6 multiplexes with the fast Ethernet on the other CPM 24. The other
7 two fast Ethernet controllers 174b, 174c provide dedicated
8 communications paths to the NPMs 14 and FPMs 22. In the
9 illustrated system of FIG. 9, the fast Ethernet controller is an
10 Intel 82559ER, fully integrated 10BASE-T/100BASE-TX LAN solution
11 combining the MAC and PHY into a single component, although such
12 embodiment is merely provided as an illustration. In the
13 illustrated system, the fast Ethernet controllers 174b, 174c
14 interface to an Ethernet switch 176 that provides fourteen
15 dedicated communication paths to the control plane for up to ten
16 FPMs 22 and two NPMs 14.

17 Data packets move into and out of the illustrated CPM 24
18 using a sixteen-bit wide 100 MHz FOCUS bus. In the illustrated
19 system, there is one full-duplex-FOCUS bus coupling each CPM 24 to
20 each NPM 14, wherein for the illustrated system of FIGs. 2 and 4
21 having dual redundant NPMs 14a, 14b, each CPM 24 couples to two
22 NPMs 14a, 14b. Serdes devices 178a, 178b convert incoming serial
23 stream data from the backplane, to parallel data for forwarding to
24 a Focus Connect device 180. Similarly, the Serdes 178a, 178b

1 convert parallel data from the Focus Connect 180 to serial data
2 before placing it on the backplane. The illustrated Focus Connect
3 180 is a switch used by the CPM 24 to direct packets to the
4 correct NPM 14. In the FIG. 9 system, packets are moved into and
5 out of the CPU memory 182 through a FPGA 184 and Input Output
6 Processor 186 that interface the Focus Connect 180 to the AGP 172.

7 Referring again to the systems of FIGs. 2 and 4, the CPMs 24
8 coordinate the different components of the switch, including the
9 NPMs and FPMs, and similarly support access to a local storage
10 device 30 that can also be referred to as a local memory device.
11 In one embodiment, the local storage device 30 can store images,
12 configuration files, and databases for executing applications on
13 the FPMs 22. For example, the local device 30 may store
14 subscriber profiles that can be retrieved for use by either the
15 NPM 14 or FPMs 22. In an embodiment, a configuration file for a
16 particular application or subscriber can be retrieved and copied
17 to multiple FPMs 22, for example, thereby providing increased
18 efficiency in a scenario wherein multiple, identically configured
19 FPMs 22 are desired. In such an embodiment, FPMs 22 may be
20 grouped for a subscriber. The local storage device 30 can be any
21 well-known memory component that may be removable or resident on
22 the CPMs 24, including but not limited to a floppy disk, compact
23 disc (CD) , digital video device (DVD), etc. In the illustrated
24 system, there is at least one local storage device for each CPM

1 24. Similarly, in the illustrated system, the local storage
2 device 30 can be divided into several partitions to accommodate
3 and protect certain processor's needs, including the processors on
4 the various FPMs 22. In one embodiment, the local storage device
5 30 can include two identical disk partitions that allow dynamic
6 software upgrades. In an embodiment, two disk partitions can
7 include identical groups of partitions that can include swap
8 partitions, common partitions for use by all processors, and
9 specific partitions for different module processors (i.e., NPMS,
10 FPMs, CPMs).

11 The illustrated CPMs 24 can also access a remote storage
12 device 32, wherein such remote storage can store services,
13 database, etc., that may not be efficiently stored in the local
14 memory device 30. The remote storage device 32 can be any
15 compilation of memory components that can be physically or
16 logically partitioned depending upon the application, and those
17 with ordinary skill in the art will recognize that the invention
18 herein is not limited by the actual memory components utilized to
19 create the remote storage device 32.

20 The FIG. 2 CPMs 24 also couple to at least one management
21 server (MS) module 28. In the illustrated embodiment, the
22 connection is a 100Base-T Ethernet connection. In the FIG. 2
23 system, the MS 28 can receive and aggregate health and status
24 information from the switch modules 14, 22, 24, wherein the health

1 and status information may be provided to the MS 28 through the
2 CPMs 24. In an embodiment wherein NPMS 14, FPMs 22, and CPMs 24
3 are redundantly provided, for example, the MS 28 can activate or
4 inactivate a particular apparatus 12 module. In the illustrated
5 embodiments, the MS 28 communicates with the apparatus 12 modules
6 through the CPM 24. In an embodiment, the MS 28 may be a PC, Sun
7 Workstation, or other similarly operational microprocessor
8 controlled device, that can be equipped with microprocessor
9 executable instructions for monitoring and controlling the
10 apparatus 12 modules. In an embodiment, the MS 28 can include an
11 executable that provides a graphical user interface (GUI) for
12 display of apparatus 12 monitoring and control information. In
13 one embodiment, the MS 28 can be a separate device from the CPM
14 24, while in another embodiment, the MS 28 functionality can be
15 incorporated into the CPM 24, for example, by utilizing a separate
16 processor on the CPM 24 for MS 28 functionality.

17 In an embodiment, the well-known Linux operating system can
18 be installed on the FPM 22 and CPM 24 processors, thereby
19 providing an open architecture that allows installation and
20 modification of, for example, applications residing on the FPMs
21 22. In the illustrated systems, the management and control of
22 applications on the switch modules can be performed using the MS
23 28. In the illustrated embodiments, the MS 28 management can be
24 performed using the CPM 24. Applications such as firewall

1 applications, etc., in the illustrated embodiments can therefore
2 be downloaded, removed, modified, transferred between FPMs 22,
3 etc. using the MS 28.

4 In an embodiment, the NPMs 14 can execute the well-known
5 VxWorks operating system on the MIPS processor and a small
6 executable on the IQ2000 processor 42. Those with ordinary skill
7 in the art will recognize that the methods and systems disclosed
8 herein are not limited to the choice of operating systems on the
9 various switch modules, and that any operating system allowing an
10 open architecture can be substituted while remaining within the
11 scope of the invention.

12 Referring now to FIG. 10, there is an illustrative block
13 diagram of a flow scheduling process 200 for the illustrated
14 systems and methods of FIGs. 2-4. As FIG. 10 indicates, for the
15 illustrated systems, the FPMs 22 can provide resource information
16 202 to the CPMs 24. The description or definition of resource
17 information can be dependent upon or otherwise defined by the
18 system configuration, and can include any information that can
19 assist in the distribution of flows between NPMs 14 and FPMs 22
20 according to a predefined or otherwise established flow scheduling
21 criteria. In an embodiment wherein it is desired that flows be
22 directed to FPMs 22 to optimize FPM 22 utilization, for example,
23 resource information can include intrinsic FPM data such as FPM
24 CPU utilization, FPM memory utilization, FPM packet loss, FPM

1 queue length or buffer occupation, etc., and those with ordinary
2 skill in the art will recognize that such metric or resource
3 information is provided merely for illustration and not
4 limitation, and other resource information can be provided to the
5 CPMs 24 from the FPMs 22 without departing from the scope of the
6 invention. Similarly, it is not necessary that any of the above-
7 mentioned illustrative resource information be provided in any
8 given embodiment of the methods and systems disclosed herein.

9 In the illustrated embodiments, FPMs 22 can be understood to
10 belong to a FPM group, where a FPM group includes FPMs 22 that are
11 configured identically, and hence a given FPM 22 is assigned to a
12 single group. In other embodiments, a given FPM 22 can be
13 assigned to various groups, for example, if groups include FPMs
14 that are capable of processing a particular application. In an
15 embodiment wherein ten FPMs 22 are present and can be referenced
16 by the numerals one through ten, respectively, and FPMs one, four,
17 five, eight, and nine are configured identically, while FPMs two
18 and three are configured identically, and FPMs six, seven, and ten
19 are configured identically, three FPM groups can be defined
20 accordingly. For a system and method according to the illustrated
21 embodiments, resource information from the FPM groups can be
22 provided to the CPM 202 in response to a query request from the
23 CPM 24; or, resource information can be provided to the CPM 24
24 automatically, for example, at scheduled intervals during which

1 the FPMs 22 are configured to transmit the resource information to
2 the CPM 24. In an embodiment, FPMs 22 from a given group can
3 transfer resource information to the CPM 24 at specified times,
4 while in another embodiment, the transfer of resource information
5 from an individual FPM 22 to CPM 24 may not be group-related or
6 dependent. In an embodiment, the transfer of resource information
7 from FPM 22 to CPM 24 can be simultaneous for all FPMs 22.

8 In the illustrated systems, for example, a FPM 22 can
9 transmit resource information to the CPM 24 at intervals of one-
10 tenth second, although those with ordinary skill in the art will
11 recognize that such timing is provided merely for illustration,
12 and the invention herein is not limited to the timing or
13 scheduling of resource information transfer between the FPMs 22
14 and the CPM 24. The illustrated system CPM 24 can be responsible
15 for parsing the FPM 22 resource information according to FPM 22,
16 and then FPM group 204. For example, for the three-FPM group
17 illustration provided previously herein, the CPM 24 can be
18 configured to identify the FPM 22 from which resource information
19 is arriving, and also identify the group to which that FPM 22
20 belongs. Those with ordinary skill in the art will recognize that
21 there are different methods for identifying the source of a data
22 message or transfer, including for example, inclusion of
23 identification in the message header, CRC, etc., and the invention

1 herein is not limited to the technique or method by which the
2 resource information can be associated to a FPM 22.

3 The illustrated CPM 24 can arrange information from the FPMs
4 22 according to FPM group, and utilize such information to compute
5 a flow scheduling vector for the FPM group 204. Although the FPMs
6 22 can provide resource information to the CPM 24 at given
7 intervals, the CPM flow schedule computation may not be
8 coincidental with such reception of information. In one
9 embodiment, the CPM 24 can update a flow schedule vector whenever
10 FPM information is obtained; however, in other embodiments, the
11 CPM 24 may average multiple updates from a given FPM 22 or FPM
12 group, before updating a flow schedule vector. For example, the
13 CPM 24 can be configured to compute a new flow schedule vector for
14 a given group at specified time intervals, or at specified FPM
15 update intervals, etc., wherein the invention herein is not
16 limited by the timing of the CPM flow schedule vector computation.

17 In an embodiment, the CPM flow schedule vector computation
18 interval can be a function of the applications residing within a
19 given FPM group. For example, if the CPM recognizes that a FPM
20 group configuration includes applications that require a given
21 time to complete, the flow schedule vector computation can be
22 performed based upon such information. In a system wherein FPM
23 group flow schedule vector computation is application dependent,

1 FPM flow schedule vectors for different FPM groups can be computed
2 independent of the other FPM groups.

3 In one embodiment, flow schedule vectors can be computed
4 based on historic intrinsic data from the FPMs. In an embodiment,
5 this historical information can be incorporated into the flow
6 schedule vector using a filter.

7 A computed flow schedule vector for a given FPM group can be
8 of varying length. For example, consider a FPM group having three
9 FPMs 22 that can be referred to as five, six, and seven. During a
10 given interval, the CPM 24 can determine that FPMs five and seven
11 are completely loaded, while FPM six is not. The vector for the
12 FPM group can be, for example, in this instance, one value that
13 identifies FPM six, and this vector may remain the same, for
14 example, until FPMs five and seven indicate a decreased loading.
15 In another illustration for this same FPM group, wherein forty
16 percent of the flows should be processed by FPM five, forty
17 percent by FPM six, and twenty percent by FPM seven, the flow
18 scheduling vector can be five values that can be arranged in
19 vector notation as: [FPM five; FPM six; FPM five; FPM six; FPM
20 seven].

21 Referring again to FIG. 10, after the CPM 24 computes a flow
22 schedule vector for a given FPM group, the CPM can transfer 206
23 the flow schedule vector to the NPMs 14. Depending upon the CPM
24 configuration, the transfer of updated flow schedule vector from

1 CPM 24 to NPM 14 may not be at the same rate as the CPM flow
2 schedule vector computation. In some embodiments, the transfer of
3 flow schedule vectors from CPM 24 to NPM 14 can be configured for
4 fixed intervals that can vary according to FPM group. In other
5 embodiments, updated flow schedule vectors for all FPM groups can
6 be transferred to the NPMs 14 at the same time. In yet another
7 embodiment, the transfer of a new flow schedule vector from CPM 24
8 to NPM 14 may only occur based upon a predetermined criteria, for
9 example, that can require a specified difference between an
10 existing flow schedule vector and a newly computed flow schedule
11 vector. Those with ordinary skill in the art will recognize that
12 the methods and systems herein are not limited by the frequency or
13 scheduling of flow schedule vector transfers between a CPM 24 and
14 NPMs 14.

15 As indicated herein, the NPMs 14 interface to subscribers
16 and/or a network, etc., and can receive flows, identify the
17 application(s) requested by the flow, and also identify which FPMs
18 22 can process the flow/request. In a system employing the flow
19 scheduling method of FIG. 10, once the NPMs 14 identify which
20 application(s) a received flow is requesting, the NPMs 14 can
21 determine a FPM group to process the flow. In one embodiment, the
22 NPMs 14 can utilize, for example, a hash table to relate a request
23 for an application or service to a particular FPM group and/or
24 flow schedule vector, although those with ordinary skill in the

1 art will recognize that there are many different techniques for
2 associating a flow or request with a processor group, and the
3 invention herein is not limited to any particular technique. The
4 NPMs can also utilize the flow schedule vector for the identified
5 FPM group to determine which FPM 22 within the identified FPM
6 group, should receive the flow/request for processing. In the
7 illustrated systems and methods wherein flow scheduling vectors
8 can be utilized, the NPMs 14 can be configured to direct flows to
9 FPMs 22 according to the flow schedule vector contents, by
10 sequentially assigning flows to FPMs 22 in the FPM order listed in
11 the respective flow schedule vector, while returning to the
12 beginning of a vector when the vector end is reached. Those with
13 ordinary skill in the art will also recognize that a flow schedule
14 vector can include pointers to FPMs, FPM identities, etc, and the
15 invention is not limited by the technique by which a particular
16 FPM is identified by the vector.

17 Those with ordinary skill in the art will recognize that the
18 FIG. 10 flow chart and associated discussion is also provided
19 merely for illustration and not limitation. For example, although
20 the flow chart discussion began with the description of the
21 resource information transferring from the FPMs 22 to the CPMs 24,
22 one with ordinary skill in the art will recognize that such
23 processing may not be the initial step in the FIG. 10 processing.
24 In an embodiment, initial flow schedule vectors can be provided by

1 the CPMs 24 to the NPMs 14, or alternately, the NPMs 14 can be
2 configured with an initial flow schedule vector for the different
3 FPM groups. The processing illustrated in FIG. 10 can thus be
4 repeated as indicated in a definite or indefinite manner, without
5 particularity for a given "beginning" or "end" of processing.

6 One advantage of the present invention over the prior art is
7 that a single architecture is disclosed with multiple processors,
8 wherein intrinsic data from the processors can be utilized to
9 generate an accurate flow scheduling vector for distributing flows
10 or data requests amongst the multiple processors.

11 What has thus been described is a method and system for
12 distributing flows between a multiple processors. The flows can
13 be received from an external source such as a network, by a front-
14 end processor that recognizes the flow and the associated request,
15 and identifies at least one internal applications processor to
16 process the request/flow. The front-end processor utilizes a flow
17 scheduling vector related to the identified applications
18 processor(s), and the flow scheduling vector can be based on
19 intrinsic data from the applications processor(s) that can include
20 CPU utilization, memory utilization, packet loss, and queue length
21 or buffer occupation. In some embodiments, applications
22 processors can be understood to belong to a group, wherein
23 applications processors within a group can be configured
24 identically. A flow schedule vector can be computed for the

1 different applications processor groups. In some embodiments, a
2 control processor can collect the intrinsic applications processor
3 data, compute the flow scheduling vectors, and transfer the flow
4 scheduling vectors to the front-end processor.

5 Although the present invention has been described relative to
6 a specific embodiment thereof, it is not so limited. Obviously
7 many modifications and variations of the present invention may
8 become apparent in light of the above teachings. For example,
9 although the illustrated systems divided the modules into various
10 components, the functionality of components may be combined into a
11 single module where appropriate, without affecting the invention.
12 Although the methods and systems herein disclosed resource
13 information transferring from the FPMS to the CPMS for computation
14 of flow scheduling vectors for further transfer to the NPMS, the
15 resource information can be transferred to the NPMS for
16 computation of the flow scheduling vectors at the NPMS.

17 Similarly, other processors can be utilized to process the
18 intrinsic resource information and compute the flow scheduling
19 vectors. Although the disclosure herein referred to a "flow
20 schedule vector", such language can be understood as indicating
21 any type of schedule of any form, and it is not necessary that the
22 schedule be in the form of a vector, queue, array, etc., as other
23 forms of scheduling or otherwise conveying order information can
24 be utilized without departing from the scope of the invention.

1 Many additional changes in the details, materials, steps and
2 arrangement of parts, herein described and illustrated to explain
3 the nature of the invention, may be made by those skilled in the
4 art within the principle and scope of the invention. Accordingly,
5 it will be understood that the invention is not to be limited to
6 the embodiments disclosed herein, may be practiced otherwise than
7 specifically described, and is to be understood from the following
8 claims, that are to be interpreted as broadly as allowed under the
9 law.

FOIA b 7 - D